Preprint
UCRL-JC-137372

Elliptic Solvers with Adaptive Mesh Refinement on Complex Geometries

B. Philip

This article was submitted to
Student Symposium 2000, Albuquerque, NM, August 10, 2000

July 24, 2000

U.S. Department of Energy

Lawrence
Livermore
National
Laboratory

Approved for public release; further dissemination unlimited

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

This is a preprint of a paper intended for publication in a journal or proceedings. Since changes may be made before publication, this preprint is made available with the understanding that it will not be cited or reproduced without the permission of the author.

This report has been reproduced
directly from the best available copy.

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information
P.O. Box 62, Oak Ridge, TN 37831
Prices available from (423) 576-8401
http://apollo.osti.gov/bridge/

Available to the public from the
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd.,
Springfield, VA 22161
http://www.ntis.gov/

OR

Lawrence Livermore National Laboratory
Technical Information Department's Digital Library
http://www.llnl.gov/tid/Library.html

Elliptic Solvers with Adaptive Mesh Refinement on Complex Geometries

Bobby Philip
Department of Applied Mathematics
University of Colorado, Boulder
bobbyp@llnl.gov

Supervisor: Dan Quinlan
Center for Applied Scientific Computation (CASC)
Lawrence Livermore National Laboratory
Livermore, CA 94550
dquinlan@llnl.gov

July 24, 2000

Abstract

Adaptive Mesh Refinement (AMR) is a numerical technique for locally tailoring the resolution of computational grids. Multilevel algorithms for solving elliptic problems on adaptive grids include the Fast Adaptive Composite grid method (FAC) and its parallel variants (AFAC and AFACx). Theory that confirms the independence of the convergence rates of FAC and AFAC on the number of refinement levels exists under certain ellipticity and approximation property conditions. Similar theory needs to be developed for AFACx. The effectiveness of multigrid-based elliptic solvers such as FAC, AFAC, and AFACx on adaptively refined overlapping grids is not clearly understood. Finally, a non-trivial eye model problem will be solved by combining the power of using overlapping grids for complex moving geometries, AMR, and multilevel elliptic solvers

This work was performed under the auspices of the U. S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under Contract No. W-7405-Eng-48.

1 Introduction

Partial differential equations arise in the modeling of various physical, chemical, and biological processes. Many of these processes exhibit the property of rapidly varying, complex behavior in small regions while varying relatively smoothly over the larger domain of interest. Other applications require simulations over widely varying length scales. For accurate numerical modeling of these processes, resolution of the computational grids must be extremely fine in the regions of complex, local variation. However, requiring the grids to have the same fine resolution over the whole computational domain is not practical. For example, three-dimensional simulations of the dynamics of the human heart with a uniformly fine grid would "require the storage and processing of about 10e10 floating point numbers per timestep" [3]. Similarly, reaction chambers in industrial boilers have characteristic dimensions ranging from meters to tens of meters, while the inlets for introducing oxidizers and reactants have dimensions on the order of millimeters [2].

Adaptive Mesh Refinement (AMR) [2], [5], [6], [7], is a numerical technique for locally tailoring the resolution of computational grids. AMR permits the addition of finer local grids to the global computational grid in an adaptive manner so as to permit locally more accurate computations or removal of the global error introduced by local singularities. Through the introduction of mesh points driven by unresolved error in the computation, typically one to two orders of improvement in efficiency can be obtained. Thus, AMR approaches are extremely attractive. It is in this context that multilevel, adaptive, iterative elliptic solvers exhibiting convergence rates independent of the number of refinement levels and scalability in a parallel environment become extremely important. Algorithms described in the following sections represent an important step in that direction. This paper is organized as follows. Section 2 provides a brief introduction to a class of multilevel, adaptive solvers - the Fast Composite Grid methods [7], [8], [9], [11]. Section 3 describes an application that models the motion of the aqueous humor and the iris in the human eye. Section 4 describes preliminary work that has been done. Section 5 presents conclusions and plans for further research.

2 Fast Adaptive Composite Grid Methods - A brief overview

In this section, three existing algorithms for the solution of elliptic equations on adaptive grids are briefly sketched: the Fast Adaptive Composite Grid method (FAC) [11]; Asynchronous FAC (AFAC) [9], [10], [11], and the AFACx variant [7]. FAC belongs to the class of multiplicative algorithms, while AFAC and AFACx are additive algorithms that allow for asynchronous processing of refinement levels (particularly important for parallel computation). Multilevel theory exists to show that FAC and AFAC have convergence rates independent of the number of levels of refinement [8]. Only the two-level theory exists in the case of AFACx [7]. Numerical experiments reported in [7] imply that AFACx exhibits the same behavior in the multilevel case.

2.1 Model Problem

Consider a linear, self-adjoint, second-order elliptic problem:

$$\begin{cases} -\sum_i \sum_j \frac{\partial}{\partial x_i} a_{ij}(x) \frac{\partial u}{\partial x_j} = f & \text{in } \Omega, \\ u = u_0 & \text{on } \partial\Omega. \end{cases} \tag{1}$$

Here, Ω is a bounded open Lipschitz polyhedral region in R^n (n = 2, 3), the matrix $\{a_{ij}(x)\}$ is symmetric and positive definite with a positive uniform lower bound for almost all x in Ω, and each $a_{ij}(x)$ is a bounded measurable function in Ω. Then, equation (1) has a unique solution [1]. Without loss of generality, we assume $u_0 = 0$.

The variational form corresponding to (1) is: Find $u \in H_0^1(\Omega)$ such that

$$a(u,v) = f(v), \quad \forall v \in H_0^1(\Omega), \tag{2}$$

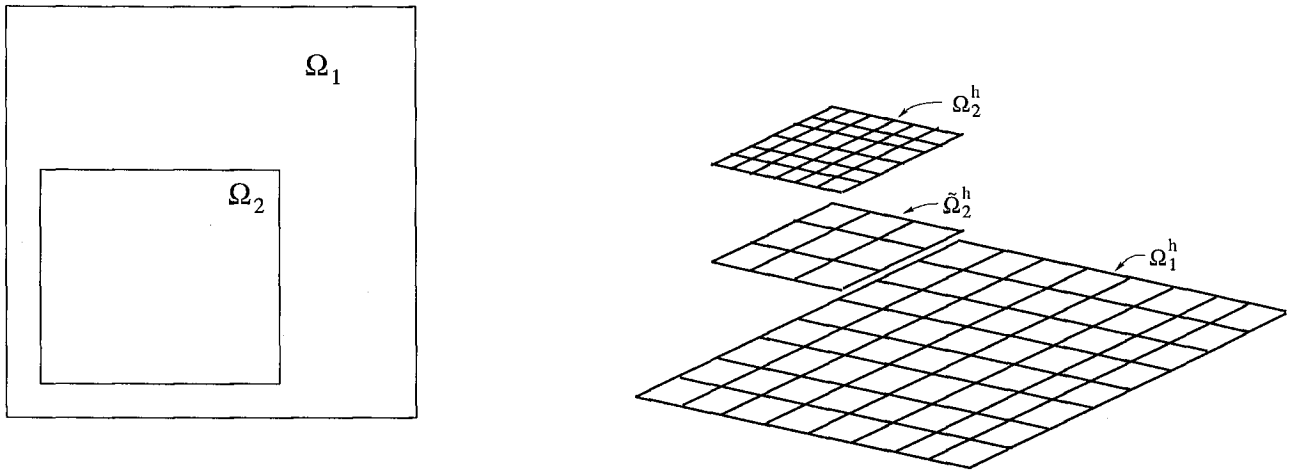

Figure 1: (a) Two nested domains Ω_1, Ω_2 (b) and the associated triangulations Ω_1^h, $\tilde{\Omega}_2^h$, Ω_2^h.

where

$$\begin{cases} a(u,v) &= \int_\Omega \sum_{ij} a_{ij}(x) \frac{\partial u}{\partial x_j} \frac{\partial v}{\partial x_i} d\Omega, \\ f(v) &= \int_\Omega f v d\Omega. \end{cases} \tag{3}$$

Note that $a(\cdot,\cdot)$ is a bounded and uniformly elliptic bilinear form on $H_0^1(\Omega) \times H_0^1(\Omega)$ and $f(\cdot)$ is a bounded linear functional on $H_0^1(\Omega)$.

To consider the solution of (1) on partially refined meshes, the following notation is introduced. Suppose there are J refinement levels associated with a nested sequence $\Omega_J \subseteq \Omega_{J-1} \ldots \subseteq \Omega_1 = \Omega$ of bounded open Lipschitz subdomains and let $\tilde{\Omega}_k = \Omega_k \bigcap \Omega_{k-1}$. Define a nested sequence of triangulations $\{\Omega_l^{h_l}\}_{l=1}^J$, where $\Omega_l^{h_l} = \{\tau_i^l\}_{i=1}^{N_l}$ is a quasi-uniform triangulation of Ω_l obtained by a regular dyadic refinement of elements $\{\tau_{i_k}^{l-1}\}_{k=1}^{M_k}, M_k \leq N_{l-1}$, which form a "coarse" triangulation of $\tilde{\Omega}_l$. For simplicity, let h_l be replaced by h when l is understood by context. Let $\tilde{\Omega}_l^h$ denote the "coarse" triangulation of $\tilde{\Omega}_l$. Let $\Omega^c = \bigcup_{k=1}^J \Omega_k^h$ denote the *composite grid* space.

For example, Figure 1(a) shows a nested sequence of 2 domains, $\Omega_2 \subset \Omega_1$. Ω_1 is triangulated in the standard finite element manner into rectangular elements to obtain Ω_1^h. This means that Ω_2 has a "coarse" triangulation (since $\Omega_2 \subset \Omega_1$) by elements of Ω_1^h. This "coarse" triangulation of Ω_2 is $\tilde{\Omega}_2^h$. The local "fine" triangulation of Ω_2, denoted by Ω_2^h, is obtained by splitting each element of $\tilde{\Omega}_2^h$ into four smaller rectangular elements as shown in Figure 1(b). $\Omega^c = \Omega_1 \bigcup \Omega_2$, is the *composite* grid shown in Figure 2(a). Figure 2(b) shows the uniform *sub-grids* that form the *composite* grid.

Define $V^k \subset H_0^1(\Omega_k), k = 1, 2, \ldots J$, to be the associated finite element space and $\tilde{V}^k = V^{k-1} \bigcap H_0^1(\Omega_k)$ to be the restricted local refinement space associated with the local refinement region Ω_k^h. $V \equiv V^c = \sum_{k=1}^J V^k$ the associated *composite* finite element space. The discrete variational problem may be formulated as: Find $u^c \in V^c$ such that

$$a(u^c, v) = f(v), \ \ \forall v \in V^c. \tag{4}$$

This is equivalent to solving the linear system

$$A^c u^c = f^c, \tag{5}$$

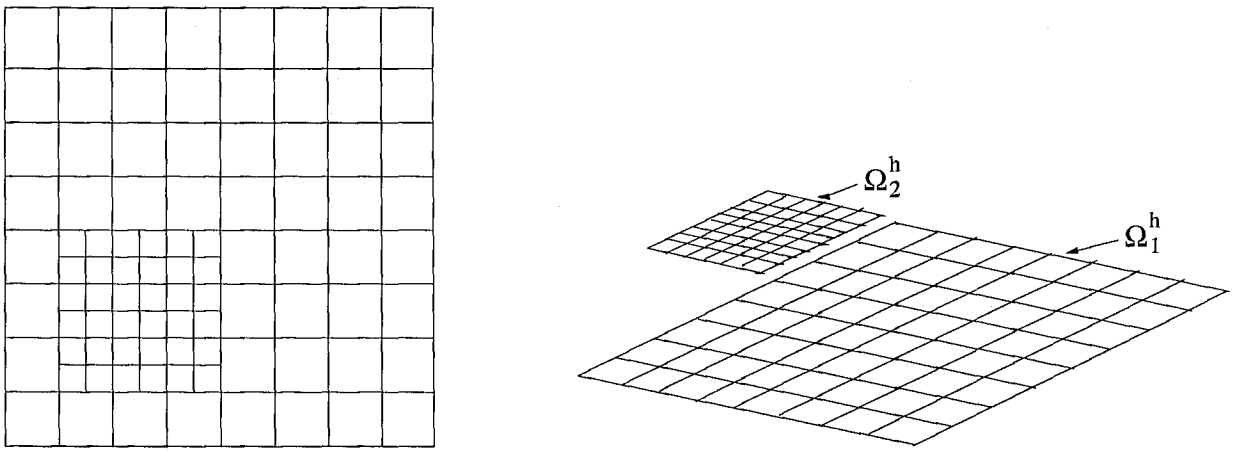

Figure 2: (a) A composite grid (b)Its composition as two uniform sub-grids

where A^c is a symmetric positive definite matrix corresponding to the linear operator defined on the composite grid.

An iterative correction scheme for solving (5) may be written as follows:

(1) $u^c \leftarrow 0$

(2) while (error in solution > tolerance) do

(i) $r^c \leftarrow f^c - A^c u^c$ (form the residual)

(ii) solve $A^c e^c = r^c$ approximately (residual equation)

(iii) $u^c \leftarrow u^c + e^c$ (correct)

The FAC, AFAC, and AFACx algorithms described in the following sections may be viewed in this residual correction manner. They are characterized by the fact that Step 2(ii) is done by solving problems on the uniform sub-grids $\Omega_i^h, i = 1, \cdots, J$ that form the composite grid Ω^c (Figure 2). The difference between these variants lies in the order of processing refinement levels and the amount of computational effort required by each.

2.2 Notation

Suppose we are given the following interlevel transfer operators:

- $I_c^k : V^c \rightarrow V^k$ and $I_k^c : V^k \longrightarrow V^c$ denote interlevel transfer operators (restriction and interpolation, respectively) between the composite grid space V^c and the space V^k defined over the k-th refinement level Ω_k^h. For example, I_k^c could be based on linear interpolation and I_c^k could be defined as the adjoint of I_k^c.
- $\tilde{I}_c^k : V^c \rightarrow \tilde{V}^k$ and $\tilde{I}_k^c : \tilde{V}^k \rightarrow V^c$ denote the interlevel transfer operators (restriction and interpolation, respectively) between the composite grid space V^c and the restricted coarse grid space $\tilde{V}^k$.
- $I_{k+1}^k : V^{k+1} \longrightarrow V^k$ and $I_k^{k+1} : V^k \longrightarrow V^{k+1}$ denote the interlevel transfer operators (restriction and interpolation, respectively) between the adjacent refinement levels Ω_k^h and Ω_{k+1}^h. The operators I_k^c and I_c^k may be considered as compositions of these operators.

- $\tilde{I}_k^k : \tilde{V}^k \rightarrow V^k$ denotes the interpolation operator from the restricted coarse grid $\tilde{\Omega}_k^h$ to the local refinement patch Ω_k^h.

Suppose we are given the following discrete linear operators:

- $A^k : V^k \rightarrow V^k$ defined by $a(u^k, v^k) = (A^k u^k, v^k), \quad \forall u^k, v^k \in V^k$,
- $\tilde{A}^k : \tilde{V}^k \rightarrow \tilde{V}^k$ defined by $a(u^k, v^k) = (\tilde{A}^k u^k, v^k), \quad \forall u^k, v^k \in \tilde{V}^k$,

where $(\cdot, \cdot)$ is the $L^2(\Omega)$ inner product. In addition, let $f^k \in V^k$ be the right-hand side associated with the discrete linear system $A^k u^k = f^k$, and $\tilde{f}^k \in \tilde{V}^k$ be the right-hand side associated with the linear system $\tilde{A}^k u^k = \tilde{f}^k$, $0 \leq k \leq J$.

Finally, suppose we are given the following relaxation operators:

- $R_k(\cdot; f^k) : V^k \rightarrow V^k$ and $\tilde{R}_k(\cdot; \tilde{f}^k) : \tilde{V}^k \rightarrow \tilde{V}^k$ denote smoothers that are mutually adjoint on $\tilde{V}^k$ ($R_k = \tilde{R}_k^*$) with respect to $(\cdot, \cdot)$.

2.3 FAC Algorithm

One iteration of FAC consists of the following basic steps:

(i). For all $k \in \{0, 1, \ldots, J\}$, set the initial guess $u^k = 0$ and compute f^k by transferring the composite grid residual to Ω_k^h: $f^k \longleftarrow I_c^k(f^c - A^c u^c)$.

(ii). Set $k = 0$, so that we start the computation on the coarsest grid, Ω_0^h.

(iii). Given the initial guess and composite grid residuals on level k, use multigrid (or, alternatively, any direct or iterative solver) to compute a correction local to that level, that is, update the approximation, u^k, to the solution of the error equation $A^k u^k = f^k$ on Ω_k^h.

(iv). If $k < J$, then:

(a) Interpolate the "solution" u^k (resulting from step (iii)) at the interface of levels Ω_k^h and Ω_{k+1}^h to supply Ω_{k+1}^h with complete boundary conditions, so that its correction equation ($A^{k+1} u^{k+1} = f^{k+1}$) is properly posed.

(b) Interpolate it also to (the interior of) Ω_{k+1}^h to act as the initial guess for u^{k+1}: $u^{k+1} \leftarrow I_k^{k+1} u^k$.

(c) $k \leftarrow k + 1$; go to (iii).

Else, if $k = J$, then interpolate all corrections (i.e., "solutions" of each level's projected composite grid residual equations) from the finest level in each region (i.e., $\Omega_k^h / \Omega_{k+1}^h$) to the composite grid Ω^c, and correct the composite grid solution: $u^c \longleftarrow u^c + \sum_{k=0}^{J} I_k^c u^k$.

As can be seen from this description, FAC is multiplicative, meaning that it can be represented as a product of linear operators. Multiplicative algorithms are sequential in nature since each operation depends on its predecessor. This sequentialness makes them less attractive in a parallel environment.

2.4 AFAC Algorithm

The FAC algorithm attempts to resolve *all* components of the solution to the composite grid error equation that "live" on a refinement level. The Asynchronous Fast Adaptive Composite Grid (AFAC) method ([9] and [10]) is based on recognizing the fact that it is sufficient to resolve components of the solution to the composite grid error equation that can *only* be represented at that refinement level. This objective is not dependent on resolving components of the solution to the error equation that "live" on coarser or finer levels, so it provides for independent level processing.

The principal step in AFAC is the computation of an approximation to the oscillatory component of the solution on each composite grid level.

Loosely speaking, one AFAC iteration consists of the following basic steps:

(i). For all $k \in \{0, 1, \ldots, J\}$, compute f^k by transferring the composite grid residual to Ω_k^h and similarly for $\tilde{f}^k$: $f^k \leftarrow I_c^k(f^c - A^c u^c)$, $\tilde{f}^k \longleftarrow \tilde{I}_c^k(f^c - A^c u^c)$.

(ii). For all $k \in \{0 \ldots J\}$, set the initial guesses $u^k = 0$ on Ω_k^h and $\tilde{u}^k = 0$ on $\tilde{\Omega}_k^h$.

(iii). For all grid levels Ω_k^h $(k \in \{0, 1, \ldots, J\})$:

(a) Use multigrid (or, alternatively, any direct or fast iterative solver) to compute a correction local to that level, that is, update the approximation u^k (resp. $\tilde{u}^k$) to the solution of the error equation $A^k u^k = f^k$ (resp. $\tilde{A}^k \tilde{u}^k = \tilde{f}^k$) on Ω_k^h (resp. $\tilde{\Omega}_k^h$).

(b) Subtract the restricted grid "solution" $\tilde{u}^k$ from the local grid "solution" u^k. This forms the "oscillatory components".

(iv). Interpolate and add the "oscillatory components" on all of Ω_k^h for all $k \in \{0, 1, \ldots, J\}$ to all finer composite levels: $u^c \leftarrow u^c + I_0^c u^0 + \sum_{k=1}^{J} (I_k^c u^k - \tilde{I}_k^c \tilde{u}_k)$.

AFAC appears to have near optimal complexity in a parallel computing environment because it allows for simultaneous processing of all levels of refinement. This is important because the solution process on each grid, even with the best solvers, dominates computational complexity. This is especially true for systems of equations where the solution process is significantly more computation intensive than the evaluation of the residuals. Coupled with multigrid processing of each level and nested iteration on the composite grids, AFAC is usually able to solve the composite grid equations in a time proportional to what it would take to solve the global grid alone. See Hart and McCormick [9] and McCormick [11] for further details.

2.5 The AFACx Algorithm

As described, AFAC removes the sequential nature of the FAC algorithm. However, it is possible to further reduce the computational effort at a given level. The basic idea is: at a given refinement level ℓ, resolve components of the solution to the composite grid residual equation that "live" *only* on that level. Hence, it is now sufficient to perform relaxation on that level and the restricted coarse grid $\tilde{\Omega}_l^h$ before computing the difference. The AFACx algorithm is an expression of these concepts.

Given the composite grid right-hand side f^c and initial approximation u^c, then one cycle of AFACx based on one relaxation sweep per level Ω_k^h and one sweep per level on $\tilde{\Omega}_k^h$ is given by the following:

(i). For all $k \in \{0, 1, \ldots, J\}$, compute f^k by transferring the composite grid residual to Ω_k^h, and similarly for $\tilde{f}^k$: $f^k \leftarrow I_c^k (f^c - A^c u^c)$, $\tilde{f}^k \leftarrow \tilde{I}_c^k (f^c - A^c u^c)$.

(ii). For all $k \in \{0 \ldots J\}$, set the initial guess $u^k = 0$ on Ω_k^h and similarly set $\tilde{u}^k = 0$ on $\tilde{\Omega}_k^h$.

(iii). For all $k \in \{0, 1, \ldots, J\}$:

(a) Relax on the restricted coarse grid equation $\tilde{A}^k \tilde{u}^k = \tilde{f}^k$ on $\tilde{\Omega}_k^h$: $\tilde{u}^k \leftarrow \tilde{R}_k(\tilde{u}^k, \tilde{f}^k)$.

(b) Interpolate the "solution" $\tilde{u}^k$ to the local fine grid Ω_k^h and relax again using the local fine grid right hand side f^k to obtain the fine grid "solution" u^k: $u^k \longleftarrow R_k(\tilde{I}_k^k \tilde{u}^k; f^k)$.

(c) Subtract the restricted grid "solution" from the local grid "solution." This forms the "oscillatory components".

(iv). Interpolate and add the "oscillatory components" on all of Ω_k^h for all $k \in \{0, 1, \ldots, J\}$ to all finer composite levels: $u^c \longleftarrow u^c + I_0^c u^0 + \sum_{k=1}^{J} (I_k^c u^k - \tilde{I}_k^c \tilde{u}_k)$.

AFACx is computationally much cheaper than AFAC because at each level, it requires only relaxing on the refinement level and the restricted coarse grid.

2.6 A comparison of FAC, AFAC, and AFACx

All three algorithms share some common advantages. Grids at the same refinement level can be processed in parallel. This is important in a distributed computing environment. The use of uniform sub-grids at a given refinement level leads to uniform stencils, simplified data structures, the ability to re-use existing global grid solvers, and facilitates supporting theory.

Though FAC exhibits a reasonable degree of parallelism, and shares the attributes given above, it is hampered by its inability to process different refinement levels asynchronously. This is a severe drawback in a distributed environment, where processors owning fine grids are forced to wait for processors owning

coarse grids (during computing the corrections on each level), and vice versa (during the computation of residuals). AFAC removes this drawback. Hence, it is able to exploit both task and data parallelism to a greater degree than FAC. However, it performs redundant calculations on each refinement level by attempting to resolve all components of the solution to the residual correction equation that exist on a given level.

AFACx was developed as a recognition of this fact. The computation at each level is now the cost of relaxing on the residual equation at that level. This represents a major reduction in computational cost. In addition, unlike FAC and AFAC, AFACx requires only an additional restricted coarse grid associated with each fine grid patch. Since, each fine grid patch was developed over a coarse grid, the existence of such a restricted patch is always guaranteed, and issues arising from successive coarsenings of overlapping grids may be avoided. It is important to note that these advantages are gained with no additional degradation of the convergence rates associated with AFAC.

3 Application problem

In this section, a prototype application problem is briefly discussed, followed by a description of the mathematical model for the physical problem. A FOSLS formulation of the problem is presented. Finally, a solution methodology that leverages the strengths of AMR on overlapping grids is proposed.

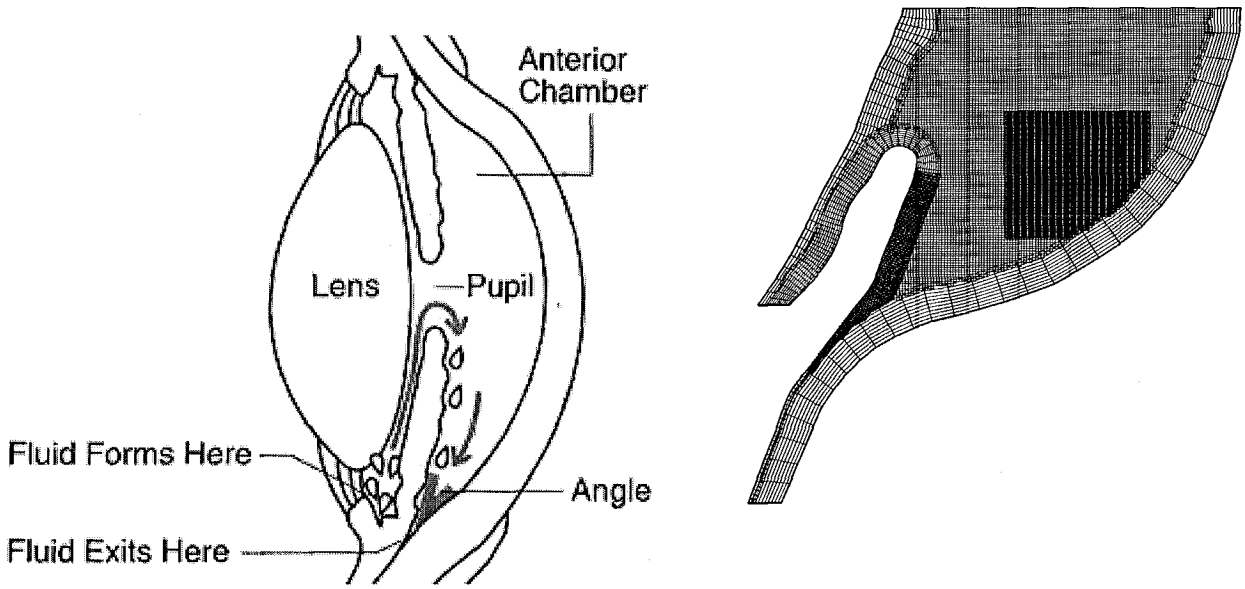

Figure 3: Eye structure and an adaptive overlapping grid.

3.1 A biological problem

The anterior chamber of the human eye is bounded by the cornea, the iris, the pupil, and the lens. The cornea is transparent, while the iris is pigmented and is responsible for the "color" of the eye. The anterior chamber is filled with a fluid called the aqueous humor, which is generated by the ciliary body and is drained out through the trabecular meshwork that is approximately 1/50th of an inch wide. The aqueous humor supplies nutrients to the lens and the cornea and removes waste generated as a result of metabolism in these tissues. The aqueous humor also provides the necessary pressure to maintain the shape of the eye. The pressure within the eye is normally 14-16 mm of Hg and is referred to as the intraocular pressure (IOP).

Glaucoma is a medical condition wherein the intraocular pressure increases, leading to the damage of the optic nerve and subsequent loss of sight. A form of glaucoma occurs when the iris rubs against the lens and sheds pigment into the aqueous humor. These particles clog the trabecular mesh and fluid outflow gets restricted, leading to increased intraocular pressure. The increased IOP in turn leads to the damage of the optic nerve and causes movement of the iris as well. This form of glaucoma is referred to as pigmentary glaucoma.

3.2 The mathematical formulation

The mathematical formulation of this problem is based on the following assumptions:

(1). The aqueous humor is a viscous, incompressible, slow moving fluid.

(2). The iris is incompressible and impermeable, but moves under the effect of increased intraocular pressure. It can be modeled using linear elasticity in the incompressible limit.

(3). The distortion/movement of the cornea and the lens due to increased IOP are negligible.

The first assumption enables us to model the flow of the aqueous humor, as Stokes flow within the fluid domain bounded by the cornea, the iris, and the lens, with the unknowns being the velocity $\mathbf{u}$ and the pressure (IOP) p. The boundary conditions are specified as velocity boundary conditions. The secretion of fluid from the ciliary body is represented as an inflow and the flow out through the trabecular mesh is represented as an outflow.

The second and third assumptions form the basis of the model for the elastic region. It is assumed that no deformation or movement of the cornea and the lens under increased IOP takes place. This should not be a major restriction and is introduced to simplify the initial mathematical model. The movement of the iris under pressure from the fluid may be modeled by the equations of linear elasticity.

The coupled elastic-fluid system has two states of interest. The initial state, where the iris is undeformed, and the final state, where the elastic region is deformed and, consequently, the fluid region is also deformed. These states shall be referred to as the "rest" and "deformed" states, respectively.

The initial rest position is of interest because it is in this configuration that the equations for linear elasticity with appropriate boundary conditions are solved. This determines the displacement vector $\mathbf{u}_e$ that eventually translates the elastic region to its final deformed position. It is in this equilibrium deformed position that the normal components of the elastic stresses on the elastic-fluid boundary will balance the normal components of the fluid stresses.

The following notation is introduced at this point:

- Ω: the open, bounded domain that contains the fluid and elastic regions. It is assumed that this domain remains invariant.
- Γ: the Lipschitz boundary of Ω, assumed to be invariant.
- $\Omega_{f,r}$, $\Omega_{e,r}$: the fluid and elastic domains in the rest states, respectively.
- $\Omega_{f,d}$, $\Omega_{e,d}$: the fluid and elastic domains in the deformed states, respectively.
- Γ_{ef}: the elastic solid-fluid interface.
- $\Gamma_{ef,r}$, $\Gamma_{ef,d}$: Γ_{ef} in the rest and deformed states, respectively.
- Γ_u: the portion of the elastic solid-fluid interface that is fixed spatially over time.
- Γ_i: the portion of the elastic solid-fluid interface that is allowed to deform over time.
- $\Gamma_{i,r}$, $\Gamma_{i,d}$: Γ_i in the rest and deformed states, respectively.

The boundary value problem for the fluid flow region may now be formulated as:

Find $\mathbf{u}_f$, p_f such that

$$\begin{cases} -\triangle \mathbf{u}_f + \nabla p_f = \mathbf{0} \text{ in } \Omega_{f,d}, \\ \nabla \cdot \mathbf{u}_f = \mathbf{0} \text{ in } \Omega_{f,d}, \\ \mathbf{u}_f = \mathbf{\Phi} \text{ on } \partial\Omega_{f,d}, \end{cases} \tag{6}$$

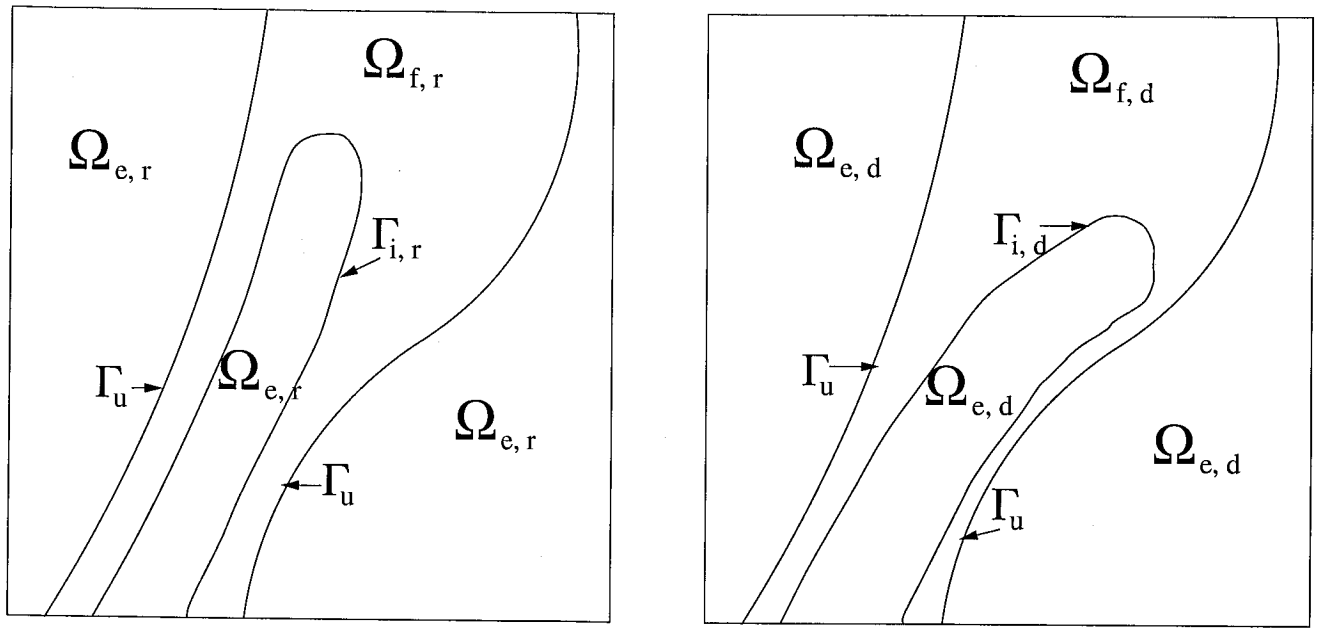

Figure 4: (a) Rest positions of regions (b) Deformed positions

together with the mean pressure condition:

$$\int_{\Omega_{f,d}} p_f dx = 0, \tag{7}$$

where $\mathbf{u}_f$ represents the fluid velocity, p_f the pressure, and Φ is a known quantity.

Similarly, the boundary value problem for the elastic region is formulated as:
Find $\mathbf{u}_e$, p_e such that

$$\begin{cases} -\triangle \mathbf{u}_e + \nabla p_e &= \mathbf{0} \text{ in } \Omega_{e,r}, \\ \nabla \cdot \mathbf{u}_e &= \mathbf{0} \text{ in } \Omega_{e,r}, \\ \mathbf{u}_e &= \mathbf{0} \text{ on } \Gamma_u, \end{cases} \tag{8}$$

where $\mathbf{u}_e$ represents the displacement, and p_e is defined in terms of $\mathbf{u}_e$.
On the fluid-solid interface $\Gamma_{i,d}$, the normal components of the fluid stresses balance the normal components of the elastic stresses. Let $\mathbf{y} = \mathbf{x} + \mathbf{u_e}(\mathbf{x})$.

$$\mathbf{n}(\mathbf{y}) \cdot [-(\nabla \mathbf{u}_f(\mathbf{y}) + (\nabla \mathbf{u}_f(\mathbf{y}))^t) + p_f(\mathbf{y})I] = \mathbf{n}(\mathbf{y}) \cdot [-(\nabla \mathbf{u}_e(\mathbf{x}) + (\nabla \mathbf{u}_e(\mathbf{x}))^t) + p_e(\mathbf{x})I)] \text{ on } \Gamma_{i,d} \tag{9}$$

4 Preliminary results

In this section, a few preliminary results are presented that demonstrate the feasibility of the approach taken.

Initially, 2D and 3D computational grids for the anterior chamber of the eye were generated. These overlapping grids (Figure 5(a) and 5(b)) were generated using the OGEN grid generation package available within the Overture framework[4]. The grids were generated by specifying splines for the boundary elements and then using sophisticated hyperbolic and elliptic grid generation algorithms available within OGEN.

Adaptive mesh refinement for curvilinear overlapping grids was implemented and is currently being tested as part of the AMR++ distribution being developed at the Center for Applied Scientific Computing, LLNL [6]. An important feature of this approach is that the mapping information used to generate the coarse grids is available to the solution method at run-time for generating the fine grid

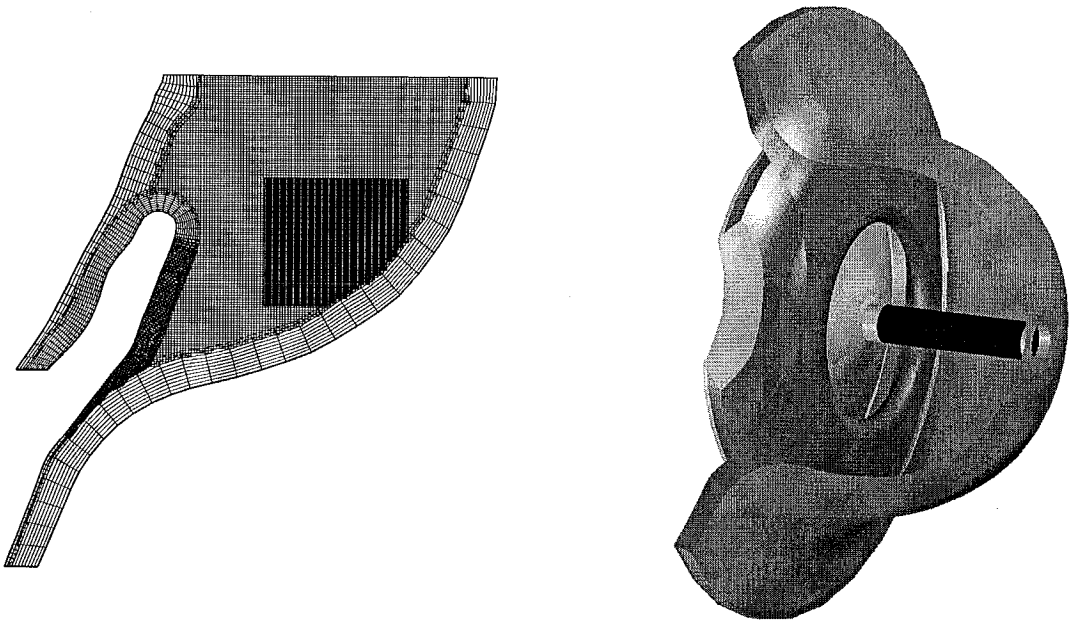

Figure 5: (a) 2D computational grid (b) section of a 3D computational grid.

patches. Hence, refinement patches mapped along boundaries resolve the boundary with a higher resolution than the coarse grid patches. This is an important benefit of the design of AMR within the Overture framework.

As a test problem, the incompressible Navier-Stokes equations were solved on the 2D overlapping grids for the fluid region within the eye (Figure 6) with viscosity $\nu = 0.005$. The OverBlown fluid flow solver for Navier-Stokes equations was used to run these tests.

FAC, AFAC, and AFACx solvers were developed within the AMR++ framework on rectangular grids to solve a Poisson problem with Dirichlet boundary conditions. The problem was run with upto ten levels of refinement. Extension of this work to curvilinear coordinate overlapping grids is currently in progress.

5 Summary

Block structured AMR techniques are an important tool in the simulation of complex processes. Multigrid based algorithms such as FAC, AFAC, and AFACx have been successfully applied to study elliptic problems in simple geometries with AMR. Algorithms such as AFACx are elegantly simple, not sacrificing the strengths of uniform global grid solvers (regular stencils, simple data structures, ease of parallelizing), while providing an efficient alternative to more complex solution methodologies.

AMR and overlapping grid techniques are powerful concepts in themselves. Combining these techniques in a seamless manner enables us to obtain meaningful results from simulations that might otherwise require either excessive computing power, or fail to resolve fine resolution features appropriately. Combining the use of AMR with overlapping grid techniques will open the door to simulating complex physical processes in realistic geometries. Together, these tools offer the promise of effectively and efficiently simulating complex real world problems. Before this becomes reality however, a host of important theoretical and practical questions arise. Providing answers to some of these questions will be the aim of this research.

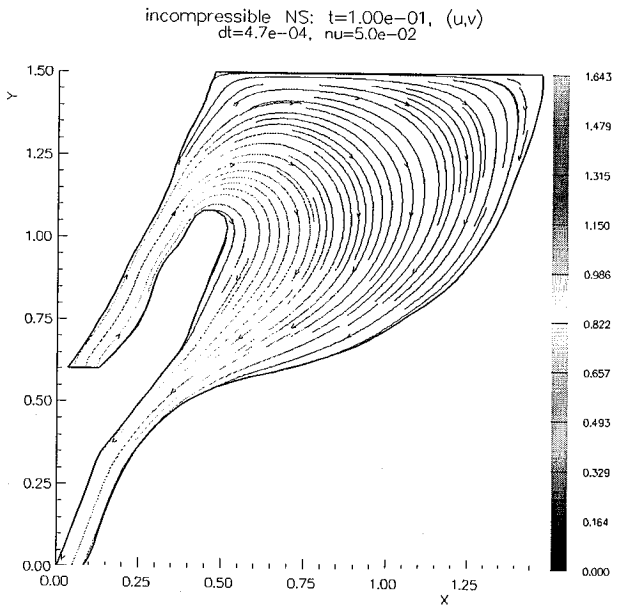

Figure 6: Contour lines for flow in the eye

References

[1] Evans, C.L., *Partial Differential Equations*, Graduate Studies in Mathematics, Vol. 19, GSM/19.

[2] Baden, S., Chrisochoides, N., Gannon, D., Norman M., (eds.), *Structured Adaptive Mesh Refinement (SAMR) Grid Methods*, The IMA Volumes in Mathematics and its Applications, Vol. 117, 2000.

[3] Cherry, E., Greenside, H., Henriquez, C., *A Space-Time Adaptive Method for Simulating Complex Cardiac Dynamics*, Physical Review Letters, Vol. 84, No. 6., Feb. 2000, pp. 1343-1346.

[4] Brown, D. L., Henshaw, W.D., and Quinlan, D., *Overture: An Object-Oriented Framework for Solving Partial Differential Equations on Overlapping Grid*, SIAM conference on Object Oriented Methods for Scientific Computing, UCRL-JC-132017, 1999.

[5] Berger, M., LeVeque R., *Adaptive Mesh Refinement using Wave-Propagation Algorithms for Hyperbolic Systems*, SIAM J. Num. Anal. 35(1998) pp. 2298-2316.

[6] Quinlan, D., *AMR++: Object Oriented Design for Adaptive Mesh Refinement*, Proceedings of High Performance Computing (HPC '98)., pp. 69-74, 1998.

[7] Quinlan, D., *Adaptive Mesh Refinement for Distributed Parallel Architectures* Ph.D thesis, University of Colorado, Denver, 1993.

[8] Cheng, H., *Iterative Solution of Elliptic Finite Element Problems on Partially Refined Meshes and the Effect of Using Inexact Solvers*, Ph.D thesis, Courant Institute of Mathematical Science, New York University, 1993.

[9] Hart, L. and McCormick, S., *Asynchronous Multilevel Adaptive Methods for Solving Partial Differential Equations: Basic Ideas*, Parallel Computing, Vol. 12, 1989, pp. 131-144.

[10] McCormick, S., Quinlan, D., *Asynchronous Multilevel Adaptive Methods for Solving Partial Differential Equations on Multiprocessors: Performance results* Parallel Computing, 12, 1989, pg. 145-156.

[11] McCormick, S., *Multilevel Adaptive Methods for Partial Differential Equations*, Frontiers in Applied Mathematics Series, No. 6, SIAM, Philadelphia, 1989.